Program 3 Report

Erik Mitchell

Student ID: 00001581305

Rank: 19, Score: 0.5490

Department of Computer Science & Engineering, Santa Clara University

CSEN 342: Deep Learning

Dr. David C. Anastasiu

March 5th, 2025

1. Introduction

The goal of this program project is to train an audio classification model that can identify the type of sounds in different audio clips. The process includes extracting features from the audio and designing a classifier which then makes classification inferences on the test dataset. This report contains the methodology, experiments, and results that were used to solve the audio classification problem.

2. Methodology

2.1. Audio Preprocessing

Several techniques were used to ensure that the audio files in the given datasets were uniformly formatted to ensure that the model can accurately train.

1. **Resampling:** Audio inputs are altered to change the sample rate. In this program, I set the sampling rate to 16,000 Hz or 16 kHz.

2. **Mixing Down:** Audio inputs are averaged to a single channel (mono channel).

3. **Cutting:** Audio inputs that are longer than 5 seconds are cut to ensure they are only 5 seconds long.

4. **Right Padding:** Audio inputs that are shorter than 5 seconds are right-padded with zero frequency values to the 5 second mark.

2.2. Feature Extraction

For this project, I have chosen to process the audio files into mel spectrograms. Mel spectrograms, displayed in Figure 1, are a way to visually represent how the frequency of an audio signal changes over time. A mel spectrogram is created by using the Fourier transform which breaks down an audio signal into its individual frequencies and the amplitude of each frequency. This transformation creates a spectrum where frequencies are displayed with respect to time and color is used to represent the decibel amplitude.

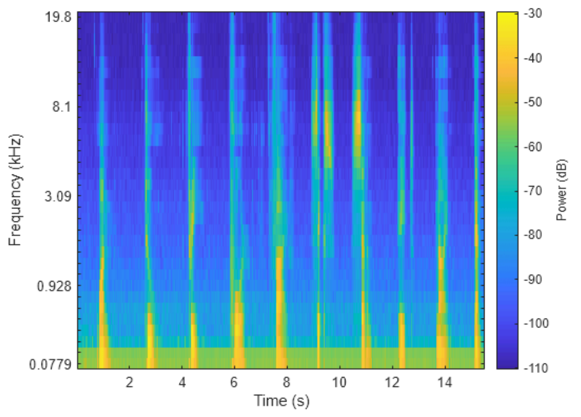

Figure 1. Mel spectrogram sample [1]

2.3. Classification Model

For this project, I have chosen to build a convolutional neural network (CNN) to analyze the audio features. The mel spectrograms that are created from the audio files are fed as inputs to the

CNN. The CNN applies filters to these spectrograms and learns to detect patterns and structures that allow it to perform classification inference on unseen audio data. The CNN used for this project consists of 4 convolutional blocks, each implementing ReLU activation and max pooling. Finally, single flattening, dropout, linear and softmax layers are used to produce probabilities for each of the 25 classes.

3. Experiments

3.1. Datasets

We are given three datasets for this project. The training set contains 350 .wav files. The validation and test sets contain 75 .wav files each. The training dataset and validation datasets each contain a labels.txt file which contains the class of each audio file.

3.2. Performance Metric

F1 score (F1) is the metric that will be used for this project. F1 measures the accuracy of a model, using both the precision and recall scores. It computes how many times a model made a correct prediction across a dataset.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

In these formulas, TP indicates true positives, FP indicates false positives, and FN indicates false negatives.

3.3 Tuning Model and Hyperparameters

Three experiments were conducted for this project, testing two different model types and two different inputs.

1. **Experiment 1:** A feedforward network processing mel frequency cepstral coefficients (MFCCs) as input. Cross entropy loss and the Adam optimizer are used. The model was trained for 100 epochs, a batch size of 32, and a learning rate of 0.001.

2. **Experiment 2:** A CNN model processing mel spectrograms as input. Cross entropy loss and the Adam optimizer are used. The model was trained for 20 epochs, a batch size of 128 is used, and a learning rate of 0.001.

3. **Experiment 3:** A modification of Experiment 2, adding a dropout layer to the CNN. The dropout rate is 0.2. In this experiment, the model was trained for 50 epochs. The rest of the parameters are the same as Experiment 2.

4. Results

4.1. Experiment 1

In the first experiment, I extracted MFCCs from the audio files. Each MFCC consists of tensors of size 20. These are fed into 4 linear layers each with ReLU activations and dropout half of the weights. This model and feature extraction achieved 0.39 F1 score.

Overall, I had difficulty getting this model to learn. Although this model could have been improved, such as adding more model parameters and less dropout, I decided to attempt a different approach as I believed that using the feed forward model would not be able to give strong classification results.

4.2. Experiment 2

In the second experiment, I decided to extract mel spectrograms from the audio files. With these visual representations of the features, I decided that implementing a CNN would be best. Mel spectrograms and the CNN structure are discussed in section 2.3. This approach trained much better than the first experiment and achieved stronger results, giving a F1 score of 0.4898.

4.3. Experiment 3

The third experiment follows the second experiment closely, besides the addition of a dropout layer in the CNN architecture and more epochs being used to train. The additional training and the dropout layer gave the model more time to refine the weights as well as reduce the complexity of the model. This resulted in an improved F1 score of 0.5490.

5. Conclusion

Audio classification is a problem that needs to be solved in many different situations. It relies on strong feature extraction and relies on a model that can properly interpret these features. Although I used a CNN with mel spectrograms, there are other models and extraction techniques that also solve this problem such as LSTMs, Transformers, Gammatone cepstral coefficients. From the lessons learned working on this project, I look forward to trying different techniques in the future.

References

[1] https://www.mathworks.com/help/audio/ref/melspectrogram.html